LA-UR-96-4011

Approved for public release; distribution is unlimited

Title: The Performance of a Fourth-Order Runge-Kutta Scheme with Cell-Edge Spatial Differences

Author(s): Catharine Vetter Alvarez
David L. Brown

Submitted to: For External Distribution

Los Alamos
National Laboratory

THE PERFORMANCE OF A FOURTH-ORDER RUNGE-KUTTA SCHEME WITH CELL-EDGE SPATIAL DIFFERENCES

CATHARINE VETTER ALVAREZ, LANL
DAVID BROWN, LANL

Introduction

In this report, we compare the phase and amplitude errors for a fourth-order Runge- Kutta method with cell-edge spatial differences (method 1, below) with those of three other methods: a fourth-order centered difference scheme (method 2, below), a second-order Runge-Kutta method with cell-edge spatial differences (method 3, below), and the Bell, Colella, Howell method (method 4, below) [2]. We also compare the performance of method 1 and a Godunov projection scheme similar to Bell, Colella, and Howell in the solution of the incompressible Navier-Stokes equations. First, a brief description of the method used to calculate the phase and amplitude errors of the methods.

Using Fourier analysis, the two-dimensional advection equation

$$\mathbf{u}_t + c\mathbf{u}_x + c\mathbf{u}_y = 0 \tag{1}$$

where $\mathbf{u} = (u, v)$, can be transformed to

$$i\omega\hat{\mathbf{u}} + i\kappa c\hat{\mathbf{u}} + i\lambda c\hat{\mathbf{u}} = 0 \tag{2}$$

where

$$\hat{\mathbf{u}}(\omega, \kappa, \lambda) = \mathbf{u}(t, x, y)e^{-i(\omega t + \kappa x + \lambda y)}. \tag{3}$$

We can solve for ω to get the dispersion relation for (1):

$$\omega = -c(\kappa + \lambda). \tag{4}$$

Now, given a difference scheme, we can follow the same procedure: Using the difference scheme, we discretize equation 1, then since

$$\mathbf{u}^n_{j,k} = \mathbf{u}(n\Delta t, j\Delta x, k\Delta y) \tag{5}$$

we can transform the equation as above. Solving for ω, we obtain a relation of the form

$$\omega\Delta t = f(\kappa\Delta x, \lambda\Delta y, c\frac{\Delta t}{\Delta x}) + ig(\kappa\Delta x, \lambda\Delta y, c\frac{\Delta t}{\Delta x}) \tag{6}$$

The phase error, ϕ, of the method is given by

$$\phi(\kappa\Delta x, \lambda\Delta y, c\frac{\Delta t}{\Delta x}) = |f(\kappa\Delta x, \lambda\Delta y, c\frac{\Delta t}{\Delta x}) + c\frac{\Delta t}{\Delta x}(\kappa\Delta x + \lambda\Delta y)| \tag{7}$$

and the amplitude error, α, is given by

$$\alpha(\kappa\Delta x, \lambda\Delta y, c\frac{\Delta t}{\Delta x}) = |g(\kappa\Delta x, \lambda\Delta y, c\frac{\Delta t}{\Delta x})|. \tag{8}$$

Mathematica was used to perform the procedure outlined above on the various methods, and to plot the results. We developed a *Mathematica* package called `Disp.m` that can be used to plot the phase and amplitude errors of the methods discussed below for any CFL number.

The Methods

No.	Method
1	Four stage Runge-Kutta with second-order cell-edge differencing
2	Four stage Runge-Kutta with fourth-order differencing
3	Two stage Runge-Kutta with second-order cell-edge differencing
4	Bell, Colella, Howell method with second-order cell-edge differencing
5	BCH method with second-order cell-edges, but without time-centering
6	Four stage Runge-Kutta with second-order differencing
7	Four stage Runge-Kutta with three-term Taylor expanded cell-edge values

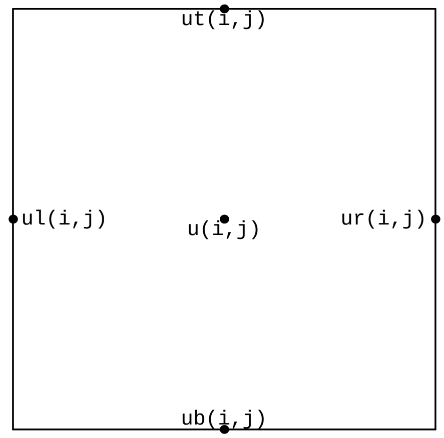

Figure 1: Notation for Edge Values

1. RK4TDo

This is the method to be compared with methods 2, 3, and 4 below. It consists of a four-stage Runge-Kutta scheme where the spatial derivatives are approximated using cell-edge values, $\tilde{\mathbf{u}}$. These are given by a two term Taylor series expansion about $\mathbf{u}^n_{i,j}$. For example,

$$\begin{aligned}
\tilde{\mathbf{u}}^{n,R}_{i+\frac{1}{2},j} &= \mathbf{u}^n_{i,j} + \tfrac{\Delta x}{2} D_{0,x}\mathbf{u}^n_{i,j} \\
\tilde{\mathbf{u}}^{n,L}_{i-\frac{1}{2},j} &= \mathbf{u}^n_{i,j} - \tfrac{\Delta x}{2} D_{0,x}\mathbf{u}^n_{i,j} \\
\tilde{\mathbf{u}}^{n,R}_{i-\frac{1}{2},j} &= \mathbf{u}^n_{i-1,j} + \tfrac{\Delta x}{2} D_{0,x}\mathbf{u}^n_{i-1,j}
\end{aligned} \tag{9}$$

The diagram above shows the positions of these variables as they are used in the Navier-Stokes code. Here, `ul(i,j)` corresponds to $\tilde{\mathbf{u}}^{n,L}_{i-\frac{1}{2},j}$, etc. To determine a single value for each edge, an upwinding procedure is used. In the case of the incompressible Navier-Stokes equations, this procedure takes the form:

$$\tilde{\mathbf{u}}^n_{i+\frac{1}{2},j} = \begin{cases} \tilde{\mathbf{u}}^{n,L}_{i+\frac{1}{2},j} & \text{if } u^n_{i,j} < 0,\ u^n_{i-1,j} < 0 \\ \tilde{\mathbf{u}}^{n,R}_{i+\frac{1}{2},j} & \text{if } u^n_{i,j} > 0,\ u^n_{i+1,j} > 0 \\ (\tilde{\mathbf{u}}^{n,L}_{i+\frac{1}{2},j} + \tilde{\mathbf{u}}^{n,R}_{i+\frac{1}{2},j})/2 & \text{otherwise} \end{cases} \tag{10}$$

For the purposes of analyzing the phase and amplitude errors, we assumed that $\Delta x = \Delta y$ and that $c > 0$ and always chose $\tilde{\mathbf{u}}^R$. The rest of the scheme is as follows:

$$\begin{aligned}
\mathbf{u}^1_{i,j} &= \mathbf{u}^n_{i,j} + \tfrac{\Delta t}{2}\mathbf{F}(\mathbf{u}^n_{i,j}) \\
\mathbf{u}^2_{i,j} &= \mathbf{u}^n_{i,j} + \tfrac{\Delta t}{2}\mathbf{F}(\mathbf{u}^1_{i,j}) \\
\mathbf{u}^3_{i,j} &= \mathbf{u}^n_{i,j} + \Delta t\mathbf{F}(\mathbf{u}^2_{i,j}) \\
\mathbf{u}^{n+1}_{i,j} &= \tfrac{(-\mathbf{u}^n_{i,j} + \mathbf{u}^1_{i,j} + 2\mathbf{u}^2_{i,j} + \mathbf{u}^3_{i,j})}{3} + \tfrac{\Delta t}{6}\mathbf{F}(\mathbf{u}^3_{i,j})
\end{aligned} \tag{11}$$

where

$$\mathbf{F}(\mathbf{u}_{i,j}) = \frac{-c}{\Delta x}(\tilde{\mathbf{u}}_{i+\frac{1}{2},j} - \tilde{\mathbf{u}}_{i-\frac{1}{2},j} + \tilde{\mathbf{u}}_{i,j+\frac{1}{2}} - \tilde{\mathbf{u}}_{i,j-\frac{1}{2}}) \tag{12}$$

2. RK4D4

In this method, the four-stage Runge-Kutta scheme above is used, but no edge values are calculated, instead the spatial derivatives are approximated by D_4:

$$\begin{aligned}
\mathbf{F}(\mathbf{u}^n_{i,j}) &= -c(D_{4,x}\mathbf{u}^n_{i,j} + D_{4,y}\mathbf{u}^n_{i,j}) \\
D_{4,x}\mathbf{u}^n_{i,j} &= \tfrac{\mathbf{u}^n_{i-2,j} - 8\mathbf{u}^n_{i-1,j} + 8\mathbf{u}^n_{i+1,j} - \mathbf{u}^n_{i+2,j}}{12\Delta x}
\end{aligned} \tag{13}$$

3. RK2TDo

This method is the same as method 1, except that a two-stage Runge-Kutta scheme is used.

$$\begin{aligned} \mathbf{u}^1_{i,j} &= \mathbf{u}^n_{i,j} + \tfrac{\Delta t}{2}\mathbf{F}(\mathbf{u}^n_{i,j}) \\ \mathbf{u}^{n+1}_{i,j} &= \mathbf{u}^n_{i,j} + \Delta t\mathbf{F}(\mathbf{u}^1_{i,j}) \end{aligned} \tag{14}$$

4. BCHDo

This corresponds to the method of Bell, Colella, & Howell.

$$\mathbf{u}^{n+1}_{i,j} = \mathbf{u}^n_{i,j} - \frac{c\Delta t}{\Delta x}(\tilde{\mathbf{u}}^{n+\frac{1}{2}}_{i+\frac{1}{2},j} - \tilde{\mathbf{u}}^{n+\frac{1}{2}}_{i-\frac{1}{2},j} + \tilde{\mathbf{u}}^{n+\frac{1}{2}}_{i,j+\frac{1}{2}} - \tilde{\mathbf{u}}^{n+\frac{1}{2}}_{i,j-\frac{1}{2}}) \tag{15}$$

where $\hat{\mathbf{u}}$ and $\tilde{\mathbf{u}}$ are the upwind values of the expansions below:

$$\begin{aligned} \tilde{\mathbf{u}}^{n+\frac{1}{2},R}_{i+\frac{1}{2},j} &= \mathbf{u}^n_{i,j} + \tfrac{(\Delta x - c\Delta t)}{2} D_{0,x}\mathbf{u}^n_{i,j} - \tfrac{c\Delta t}{2\Delta x}(\hat{\mathbf{u}}^n_{i,j+\frac{1}{2}} - \hat{\mathbf{u}}^n_{i,j-\frac{1}{2}}) \\ \hat{\mathbf{u}}^{n,T}_{i,j+\frac{1}{2}} &= \mathbf{u}^n_{i,j} + \tfrac{(\Delta x - c\Delta t)}{2} D_{0,y}\mathbf{u}^n_{i,j} \end{aligned} \tag{16}$$

5. ModDo

This method is the same as method 4 except that

$$\hat{\mathbf{u}}^{n,T}_{i,j+\frac{1}{2}} = \mathbf{u}^n_{i,j} + \frac{\Delta x}{2} D_{0,y}\mathbf{u}^n_{i,j} \tag{17}$$

6. RK4Do

This is the four-stage Runge-Kutta scheme in method 1, with the spatial derivatives approximated by D_0.

7. RK4T3

This method is the same as method 1 except that a three-term Taylor series expansion is used to obtain the edge values:

$$\tilde{\mathbf{u}}^{n,R}_{i+\frac{1}{2},j} = \mathbf{u}^n_{i,j} + \frac{\Delta x}{2}(\mathbf{u}^n_{i+1,j} - \mathbf{u}^n_{i-1,j}) + \frac{(\Delta x)^2}{8} D_{+,x}D_{-,x}\mathbf{u}^n_{i,j} \tag{18}$$

Phase and Amplitude Errors

Figure 2 shows the phase errors for methods 1–4; the three Runge-Kutta methods were plotted at CFL = 1.8, while the Bell, Colella, Howell method was plotted at CFL = 0.9. RK4TDo has less phase error than RK2TDo, as one would expect. However, RK4D4 and BCHDo both have less phase error than RK4TDo.

Figure 3 shows the amplitude errors for methods 1–4; again, the three Runge-Kutta methods were plotted at CFL = 1.8, while the Bell, Colella, Howell method was plotted at CFL = 0.9. The amplitude errors for the three Runge-Kutta methods are virtually identical (with RK2TDo slightly higher), while that for BCHDo is much less.

The other methods discussed above can also be plotted using the *Mathematica* package `Disp.m`. The function names for the methods are given above. For example, to plot the diagonal of the phase error for RK4Do, plot the function `ReDiagRK4Do[`$cfl, \kappa\Delta x$`]`. Surface plots can also be generated by plotting the two-dimensional functions, e.g. `ReRK4Do[`$cfl, \kappa\Delta x, \lambda\Delta y$`]`. The plots above were made using `PlotReBW[`cfl, a, b`]`, where $0 < \kappa\Delta x < a$ and $0 < \omega\Delta t < b$. `PlotImBW[`cfl, a, b`]` was used to plot the amplitude errors.

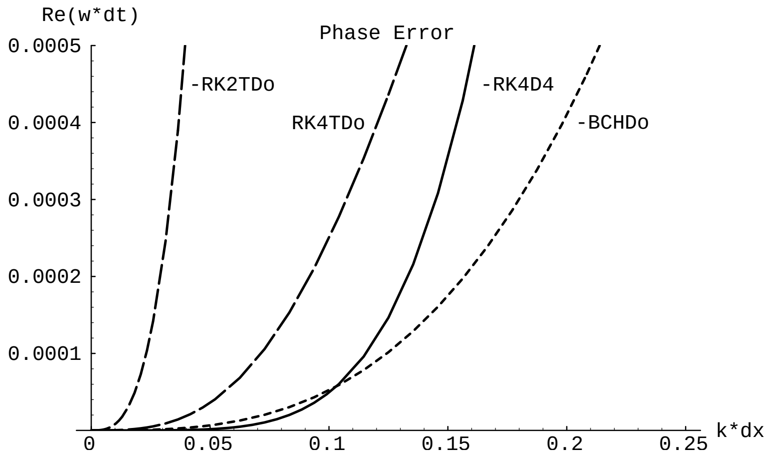

Figure 2: Diagonals of Phase Errors for Methods 1–4

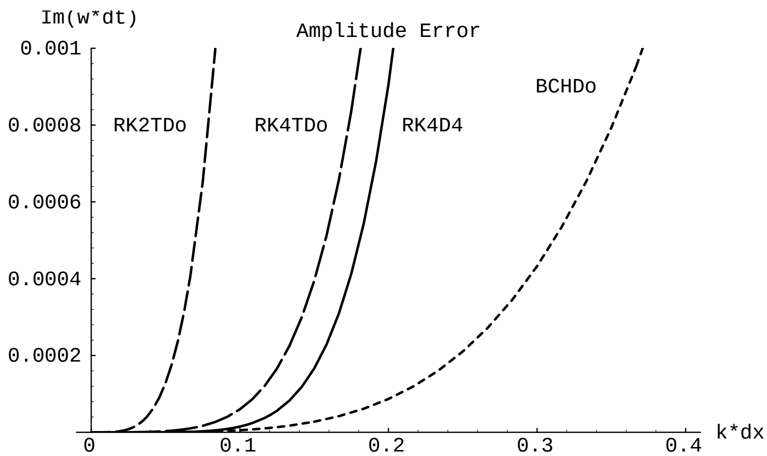

Figure 3: Diagonals of Amplitude Errors for Methods 1–4

Application to Incompressible Navier-Stokes Equations

We have adapted W. J. Rider's projection code [3], `plmins`, to use RK4TDo to solve the incompressible Navier-Stokes equations. We tested four different methods of calculating the predictor:

1. Incremental Projection Form

In this method, the old pressure is used in the predictor at each Runge-Kutta stage, i.e.

$$\begin{aligned}
\mathbf{u}^1 &= \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^n - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^n - \nabla p^n) \\
\mathbf{u}^2 &= \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^1 - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^1 - \nabla p^n) \\
\mathbf{u}^3 &= \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^2 - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^2 - \nabla p^n) \\
\mathbf{u}^* &= \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^3 - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^3 - \nabla p^n)
\end{aligned} \tag{19}$$

Then $\mathbf{u}^*$ is decomposed into the sum of its divergence free part, $\mathbf{u}^{n+1}$ and the gradient of a scalar potential, ϕ. For details, see [1].

$$\mathbf{u}^* = \mathbf{u}^{n+1} + \nabla\phi \tag{20}$$

Since $\mathbf{u}^*$ is divergence free, we have

$$\Delta\phi = \nabla \cdot \mathbf{u}^* \tag{21}$$

We solve for ϕ, and use the result to update the velocity and pressure.

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \nabla\phi \tag{22}$$

$$p^{n+1} = p^n + \frac{\phi}{\Delta t} \tag{23}$$

2. Pressure Projection Form

In this method the pressure is not used in the predictor step, i.e.

$$\begin{aligned}
\mathbf{u}^1 &= \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^n - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^n) \\
\mathbf{u}^2 &= \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^1 - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^1) \\
\mathbf{u}^3 &= \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^2 - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^2) \\
\mathbf{u}^* &= \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^3 - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^3)
\end{aligned} \tag{24}$$

So the potential, ϕ, instead of being the change in the pressure, is the new pressure itself.

$$p^{n+1} = \frac{\phi}{\Delta t} \tag{25}$$

3. Intermediate Projection Form

In this method, a pressure projection is done at each Runge-Kutta stage:

$$\begin{aligned}
&\mathbf{u}^{1,*} = \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^n - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^n) \\
&\Delta\phi^1 = \nabla\cdot\mathbf{u}^{1,*} \\
&\mathbf{u}^1 = \mathbf{u}^{1,*} - \nabla\phi^1
\end{aligned} \tag{26}$$

$$\begin{aligned}
&\mathbf{u}^{2,*} = \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^1 - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^1) \\
&\Delta\phi^2 = \nabla\cdot\mathbf{u}^{2,*} \\
&\mathbf{u}^2 = \mathbf{u}^{2,*} - \nabla\phi^2
\end{aligned} \tag{27}$$

$$\begin{aligned}
&\mathbf{u}^{3,*} = \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^2 - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^2) \\
&\Delta\phi^3 = \nabla\cdot\mathbf{u}^{3,*} \\
&\mathbf{u}^3 = \mathbf{u}^{3,*} - \nabla\phi^3
\end{aligned} \tag{28}$$

$$\mathbf{u}^* = \mathbf{u}^n + \frac{\Delta t}{2}(\nu\Delta\mathbf{u}^3 - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^3) \tag{29}$$

The final projection is the same as in the incremental projection method.

4. Hybrid Form

Here, the old pressure is used in the first three Runge-Kutta stages, and a pressure projection is done on the final stage.

$$\begin{aligned}
\mathbf{u}^1 &= \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^n - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^n - \nabla p^n) \\
\mathbf{u}^2 &= \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^1 - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^1 - \nabla p^n) \\
\mathbf{u}^3 &= \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^2 - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^2 - \nabla p^n) \\
\mathbf{u}^* &= \mathbf{u}^n + \tfrac{\Delta t}{2}(\nu\Delta\mathbf{u}^3 - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^3)
\end{aligned} \tag{30}$$

The potential, ϕ, turns out to be a combination of the new pressure and a correction to the old pressure so that

$$p^{n+1} = \frac{5}{6}p^n + \frac{\phi}{\Delta t} \tag{31}$$

Limiters

There are three options for computing spatial derivatives in the cell-edge calculation. They can be calculated with D_0 or with second- or fourth-order limited difference approximations. The second-order limited difference is given by

$$\delta^2(u)_{i,j} = \mathrm{sgn}(D^c(u)_{i,j}) \max[0, \min(|D^c(u)_{i,j}|, 2\,\mathrm{sgn}(D^c(u)_{i,j})\, D^l(u)_{i,j}, 2\,\mathrm{sgn}(D^c(u)_{i,j})\, D^r(u)_{i,j})] \tag{32}$$

where

$$\begin{aligned} D^l(u)_{i,j} &= u_{i,j} - u_{i-1,j} \\ D^r(u)_{i,j} &= u_{i+1,j} - u_{i,j} \\ D^c(u)_{i,j} &= (u_{i+1,j} - u_{i-1,j})/2 \end{aligned} \tag{33}$$

so that $(u_x)_{i,j} \approx \delta^2(u)_{i,j}/\Delta x$.

The fourth-order limited difference is given by

$$\begin{aligned} \delta^4(u)_{i,j} = \mathrm{sgn}(D^c(u)_{i,j}) \max[0, \min(|\frac{4\, D^c(u)_{i,j}}{3} - \frac{(\delta^2(u)_{i+1,j} + \delta^2(u)_{i-1,j})}{6}|, \\ 2\,\mathrm{sgn}(D^c(u)_{i,j})\, D^l(u)_{i,j}, 2\,\mathrm{sgn}(D^c(u)_{i,j})\, D^r(u)_{i,j})] \end{aligned} \tag{34}$$

As before, $(u_x)_{i,j} \approx \delta^4(u)_{i,j}/\Delta x$.

The results presented below were calculated using the fourth-order limited difference approximation.

Results

The methods were compared for speed on a doubly periodic shear layer problem using the input parameters below. Each was run at its maximum CFL.

Variable	Value
X-Length	1
Y-Length	1
viscous	F
problem	3
second order limiters	F
fourth order limiters	T
endtime	1.0
jump width	30
velocity perturbation	0.05
number of modes	-1

The table below gives the time taken by each method at its maximum CFL. The four Runge-Kutta methods slower than the Godunov projection method, even at maximum CFL. This is due to the time-step restrictions placed on these methods. For the Runge-Kutta methods,

$$\Delta t = \frac{\mathrm{CFL}}{2.55(||u||_\infty/\Delta x + ||v||_\infty/\Delta y)} \tag{35}$$

while for the Godunov projection method, the factor 2.55 is unnecessary.

Method	CFL	Time
Incremental	2.3	59.814 s
Pressure	2.3	59.461 s
Intermediate	2.3	141.114 s
Hybrid	2.3	59.801 s
Godunuov	1.0	45.110 s

Figures 5–9 show the flow calculated by each method. Figure 4 gives the L^2-norm of the divergence as time progresses.

How to Run VPLMINS

1. Select grid size by changing NX and NY in `param.h`
2. Recompile if necessary
3. Type `vplmins`
4. Select options using `t` or `f`
5. Use `mplot` to process the output file, `plm#.dat`
6. Plot the results using `plotmtv`

The code is only set up for square grids so *X-Length* should always equal *Y-Length* and NX should equal NY.

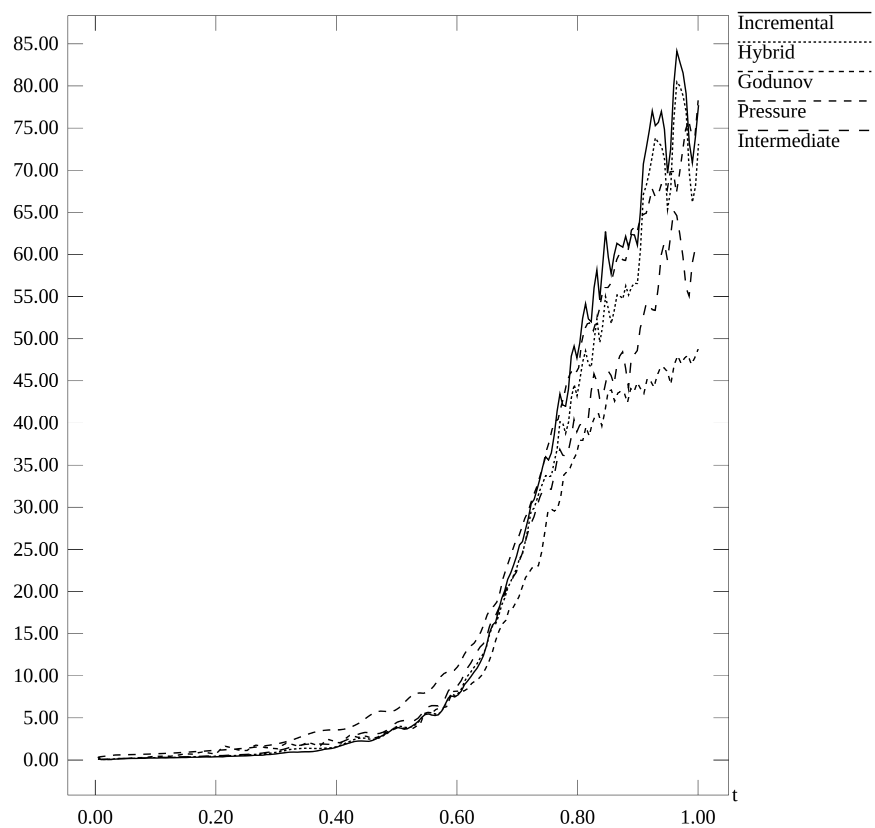

Figure 4: L^2-norm of Divergence

References

[1] D. L. Brown, *Solution methods for the incompressible Navier-Stokes equations*, report for internal distribution, Los Alamos National Laboratory, (1993).

[2] J. B. Bell, P. Colella, L. H. Howell, *An efficient second-order projection method for viscous incompressible flow*, in Proceedings of the Tenth AIAA Computiational Fluid Dynamics Conference, AIAA, June 1991, pp. 360–367.

[3] W. J. Rider, *An algorithm for accurately computing solutions to incompressible flows*, Los Alamos National Laboratory, (1994).

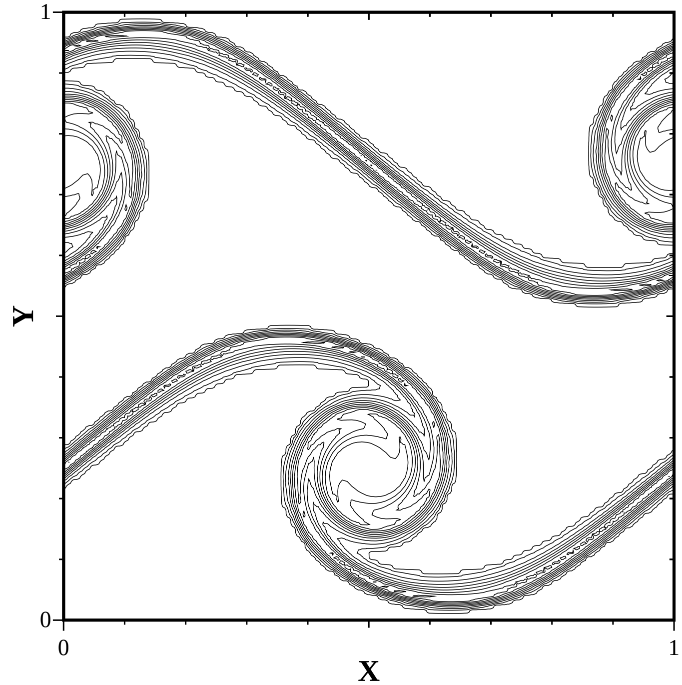

Figure 5: Incremental Projection Method, Vorticity at t = 1.0

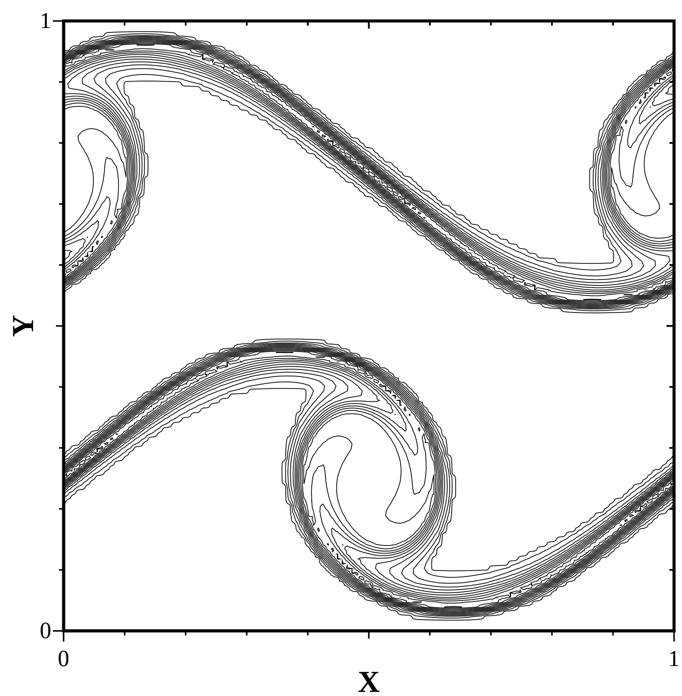

Figure 6: Pressure Projection Method, Vorticity at t = 1.0

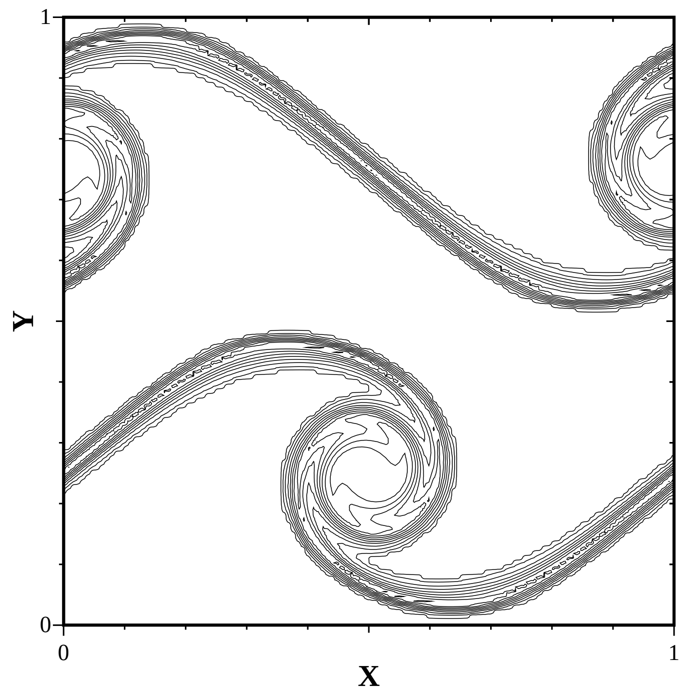

Figure 7: Intermediate Projection Method, Vorticity at t = 1.0

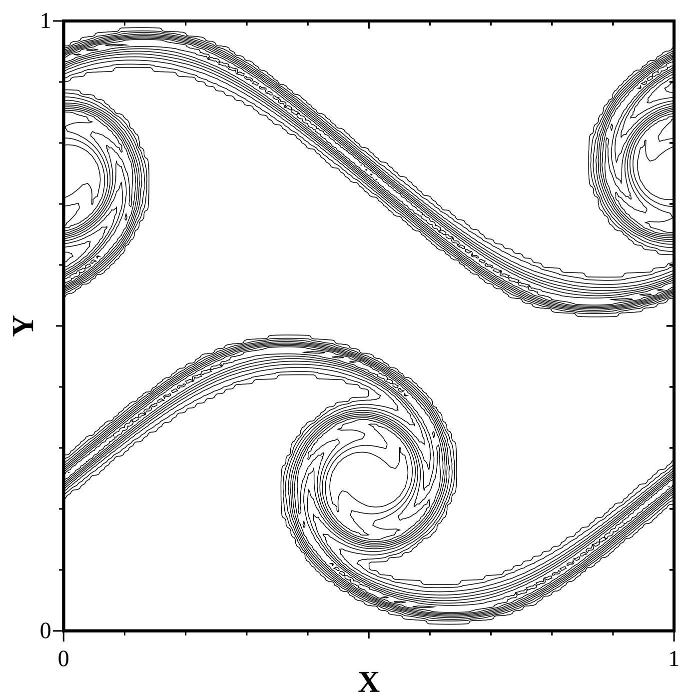

Figure 8: Hybrid Method, Vorticity at t = 1.0

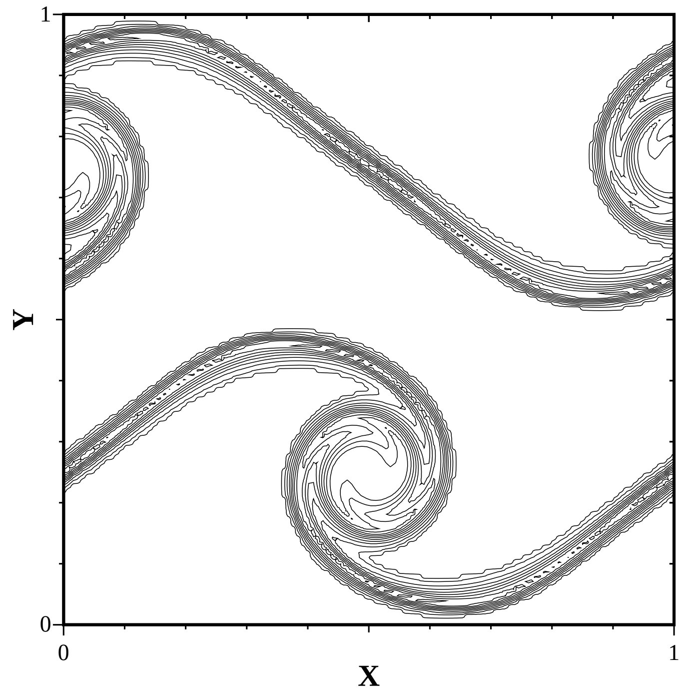

Figure 9: Godunov Projetion Method, Vorticity at t = 1.0